IN THE SPECIFICATION

Please replace the paragraph at page 1, lines 20-26, with the following rewritten paragraph:

~~Date~~ Data handled in a programming language are classified in various data types depending on the available values of the data, the types of operations, and the like. For example, in addition to a basic data type such as an integer type, a real number type, a single-precision type, and a double-precision type, a type using a structure constituted by the same types or different types as a unit is known.

Please replace the paragraph at page 2, line 30, to page 3, line 2, with the following rewritten paragraph:

On the other hand, when an abstract data type is applied to a multiphase type, a function handling data of the abstract data type is particularly called a multiphase type member function. A definition indicated by 109 in Fig. 3 defines ~~a multiphase~~ a multiphase type member function.

Please replace the paragraph at page 3, lines 8-11, with the following rewritten paragraph:

The conventional program language processing system is ~~comprises~~ comprised by a software driver 201, a preprocessor 203, a language processor 204, and a linker 205 as shown in Fig. 4 to generate an object program from a source program.

Please replace the paragraph at page 3, lines 17-26, with the following rewritten paragraph:

More specifically, as shown in Fig. 5, when the source programs 202 are given to the corresponding preprocessors 203, respectively, preprocessing such as translation is executed, and preprocessed programs 207 each having a format which can be interpreted by the language processor 204 of the following stage are obtained. The language ~~processors~~ processor 204 compiles the preprocessed programs 207, respectively, and ~~output~~ outputs relocatable objects 208 to the linker 205 as files. The linker 205 links the input relocatable object files, thereby obtaining an object program 206.

Please replace the paragraph at page 15, lines 26-28, with the following rewritten paragraph:

Fig. 13 is a concept diagram showing the flow of processes by the program language processing system according to the first embodiment.

Please replace the paragraph at page 17, lines 8-22, with the following rewritten paragraph:

The optimizing processor 4 scans the preprocessed program f0 inputted at this time to execute optimization of a code according to the first embodiment, and searches for a multiphase type definition existing in the input program f0 (step S14). If the multiphase type definition does not exist in the source program f0 (NO in step S14), the processing shifts to the searching process for a multiphase type instance in step S18 (to be described later). On the other hand, if the multiphase type definition exists in the program f0 (YES in step S14), it is checked on the basis of the definition information in the multiphase type definition table 4a whether the multiphase type definition has been registered or not (step S15). If the

multiphase type definition is a definition which ~~does~~ has not ~~have~~ been registered (NO in step S15), the definition information is registered into the multiphase type definition tables 4a (step S16).

Please replace the paragraph at page 18, line 28, to page 19, line 11, with the following rewritten paragraph:

After the scanning of the multiphase type definition is completed by steps S14 to S17 described above, the optimizing processor 4 searches for the presence/absence of an instantiation request of a multiphase type definition using a multiphase type in the source program f0 (step S18). If the instantiation request of the multiphase type definition exists (YES in step S18), it is checked on the basis of entries in, e.g., the multiphase type definition table 4a whether the type of a multiphase type definition to be instantiated has existed (step S19). If the definition is a multiphase type definition ~~which does~~ has not ~~have~~ been instantiated (NO in step S19), the instance of the multiphase type (definition) is generated (step S20). On the other hand, if the multiphase type definition has been already instantiated (YES in step S19), the instantiated definition is a duplicate, and therefore, the instance of the multiphase type is not generated (step S21).

Please replace the paragraph at page 19, lines 12-30, with the following rewritten paragraph:

Here, the details of the generation processing for the multiphase type instance in step S20 will be described with reference to Figs. 16A and 16B. When the optimizing processor 4 detects a multiphase type instance generation request 45 (shown in Fig. 16A) using, e.g., an int type, the optimizing processor 4 inquires of the multiphase type definition table 4a (shown in Fig. 15B) whether the instance of the multiphase type definition has been generated or not.

At this time, if the instance ~~does~~ has not ~~have~~ been generated, the optimizing processor 4 generates the instance of the multiphase type definition of an int type indicated by 43, and registers information (e.g., may be held in the multiphase type definition table 4a in Fig. 15B as an instantiation flag) representing that the instance has been generated into the multiphase type definition table 4a. In this case, the name of the multiphase type is converted into a unique name in the program, and all the type names of the multiphase type instance generation request 45 using the int type and the like ~~is~~ are converted into a unique name as indicated by 48.

A8

Please replace the paragraph at page 21, lines 19-22, with the following rewritten paragraph:

Figs. 17A and 17B are ~~diagram~~ diagrams for explaining an exemplary registration of a multiphase type function definition into a multiphase type definition table 4b according to the second embodiment.

A9

Please replace the paragraph at page 21, line 30, to page 22, line 6, with the following rewritten paragraph:

If the definition is a multiphase type function definition which has not been registered (NO in step S15), a type name "max" 52 of the detected multiphase type is registered in Fig. 14).

A10

Please replace the paragraph at page 23, lines 2-11, with the following rewritten paragraph:

If the instance of the multiphase type function definition ~~does~~ has not ~~have~~ been generated, the optimizing processor 4 generates the instance of a multiphase type function

A11

definition of an int type indicated by 62 (shown in Fig. 18C), and the instantiation information (e.g., as a usage flag) representing the instance has been generated is registered in the multiphase type definition table 4b. In this case, the name of the multiphase type function is converted into a unique name in a program, and all function names using the function and indicated by 64, 66, and the like are converted into the unique name.

A1

Please replace the paragraph at page 24, lines 16-23, with the following rewritten paragraph:

Figs. 19A and 19B are ~~diagram~~ diagrams showing a registration of a multiphase type member function definition into a multiphase type definition table 4c according to the third embodiment of the present invention. Fig. 20 is a flow chart showing a procedure of optimizing processing of a multiphase type member function according to the third embodiment. This processing is executed in generation processing for a multiphase type instance (definition) in step S20 in Fig. 14.

A2

Please replace the paragraph at page 25, lines 4-10, with the following rewritten paragraph:

Thereafter, the optimizing processor 4 detects a multiphase type instance generation request 76 (step S18 in Fig. 14), and the optimizing processor 4 inquires of the multiphase type definition table 4c. If the instance of the multiphase type definitions ~~does~~ has not ~~have~~ been generated (NO in step S19 in Fig. 14), the instance (definition) of the multiphase type definition is generated (step S20).

A3

Please replace the paragraph at page 25, line 32, to page 26, line 7, with the following rewritten paragraph:

A14

When the optimizing processor 4 detects, e.g., the multiphase type instance generation request 76 of the int type (Fig. 21A), the optimizing processor 4 inquires of the multiphase type definition table 4c. If the instance of the multiphase type definition ~~does~~ has not ~~have~~ been generated, the multiphase type member function which is to be actually generated is determined on the basis of usage information (member function usage flag).

Please replace the paragraph at page 26, lines 15-24, with the following rewritten paragraph:

A15

Similarly, when a double-type multiphase type instance generation request 77 is detected, if the instance of the multiphase type definition ~~does~~ has not ~~have~~ been generated, it is determined referring to the multiphase type definition table 4c on the basis of usage information of the multiphase type member function that the multiphase type member function 75 actually generated is only one member function. According to this determination, a member function whose instance is to be actually generated for a double-type multiphase type definition 80 is only one member function indicated by 81.